대 한 민 국 특 허 청
KOREAN INDUSTRIAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Industrial Property Office.

출 원 번 호 : 1998년 특허출원 제23335호
Application Number

출 원 년 월 일 : 1998년 6월 22일
Date of Application

출 원 인 : 삼성전자 주식회사
Applicant(s)

1999 년 6 월 4 일

특 허 청
COMMISSIONER

특허출원서

【출원번호】 98-023335
【출원일자】 1998/06/22
【발명의 국문명칭】 다중 뱅크 스택 구조를 위한 제어 회로
【발명의 영문명칭】 CONTROL CIRCUIT FOR MULTI-BANK STACK STRUCTURE
【출원인】
【국문명칭】 삼성전자 주식회사
【영문명칭】 SAMSUNG ELECTRONICS CO., LTD.
【대표자】 윤종용
【출원인코드】 14001979
【출원인구분】 국내상법상법인
【전화번호】 02-760-6048
【우편번호】 442-370
【주소】 경기도 수원시 팔달구 매탄동 416번지
【국적】 KR
【대리인】
【성명】 임창현
【대리인코드】 H361
【전화번호】 02-3453-7631
【우편번호】 135-080
【주소】 서울특별시 강남구 역삼동 827-53 삼호빌딩 3층
【발명자】
【국문성명】 김홍규
【영문성명】 KIM, HONG KYU
【주민등록번호】 710222-1052411
【우편번호】 449-900
【주소】 경기도 용인시 기흥읍 농서리 산 24번지
【국적】 KR
【취지】 특허법 제42조의 규정에 의하여 위와 같이 출원합니다.
대리인 임창현 (인)
【수신처】 특허청장 귀하
【수수료】
【기본출원료】 20 면 29,000 원
【가산출원료】 0 면 0 원
【우선권주장료】 0 건 0 원
【심사청구료】 0 항 0 원
【합계】 29,000 원
【첨부서류】 1. 요약서, 명세서(및 도면) 각 1통
2. 출원서 부본, 요약서, 명세서(및 도면)을 포함하는 FD부본 1통
3. 위임장(및 동 번역문)

【요약서】

【요약】

폭(width)이 M 비트이고 최대 2^N 개를 저장할 수 있는 스택(stack)을 K 개의 다중 뱅크(multi-bank) 구조로 사용하기 위한 본 발명의 제어 회로는, 상기 스택의 현재 주소값을 저장하기 위한 제 1 스택 포인터와, 상기 K 개의 다중 뱅크 각각의 현재 주소 값을 저장하기 위한 K 개의 제 2 스택 포인터들과, 상기 제 1 스택 포인터에 저장된 주소 값의 상위 X 비트에 1을 가/감산하여 출력하는 가감산기와, 상기 제 1 스택 포인터에 저장된 주소 값과 상기 가감산기의 연산 결과를 입력하여 선택적으로 상기 제 1 스택 포인터로 제공하는 멀티플렉서와, 상기 K 개의 제 2 스택 포인터들에 각각 대응하고, 상기 제 1 스택 포인터에 저장된 주소 값과 상기 가/감산 수단의 연산 결과를 입력하여 둘 중의 하나를 선택적으로 상기 제 2 스택 포인터들로 제공하는 K 개의 멀티플렉서들을 포함한다. 상기 가감산기, 멀티플렉서들은 현재 수행되는 명령어의 종류에 따라 제어되고, 상기 제 1 스택 포인터의 하위 N - X 비트는 현재 주소 값에 대응하는 뱅크를 표시하고, 현재 수행되는 명령어에 의해 M 비트의 데이터/어드레스가 상기 스택에 저장(push)되거나 인출(pop)될 때 토글 된다. 본 발명의 다중 뱅크 구조의 스택을 제어하기 위한 회로는 종래의 제어 회로보다 회로 구성이 간단하므로 하드웨어 구성 면적이 줄어든다. 또한, 소비 전력이 감소된다.

【대표도】

도 5

【명세서】

【발명의 명칭】

다중 뱅크 스택 구조를 위한 제어 회로(CONTROL CIRCUIT FOR MULTI-BANK STACK STRUCTURE)

【도면의 간단한 설명】

도 1은 종래의 다중 뱅크 스택을 제어하기 위한 회로를 블록적으로 보여주고 있는 블록도;

도 2는 일반적인 스택의 구조를 블록적으로 보여주고 있는 블록도;

도 3은 도 2에 도시된 스택을 다중 뱅크 구조로 분할하여 사용할 때 스택의 구조를 블록적으로 보여주는 블록도;

도 4a 및 도 4b는 도 3에 도시된 다중 뱅크 구조의 스택을 각각 뱅크 0과 뱅크 1로 나누어 도시한 도면;

도 5는 본 발명의 바람직한 실시예에 따른 다중 뱅크 구조의 스택을 제어하기 위한 회로를 블록적으로 도시한 도면이다.

도면의 주요 부분에 대한 부호의 설명

100, 200 : 스택　　10, 210 : 스택 포인터

20 : 가감산기　　40 : 증가기

30, 50, 60, 214, 216, 218 : 멀티플렉서

70, 220 : 뱅크1 스택 포인터　　80, 230 : 뱅크0 스택 포인터

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

본 발명은 마이크로 컨트롤러 유닛(microcontroller unit), 디지털 신호 처리기(digital signal processor) 등에서 사용되는 스택(stack)을 제어하기 위한 회로에 관한 것으로, 좀 더 구체적으로는 다중 뱅크(multi-bank) 구조의 스택을 효율적으로 제어하기 위한 회로에 관한 것이다.

마이크로 컨트롤러 유닛(Micro Controller Unit; MCU)이나 디지털 신호 처리기(Digital Signal Processor; DSP) 코어와 같은 프로그래머블 디바이스(programmable Device)에서 널리 사용되는 스택(stack)은 크게 나누어 2가지 형태로 구현 가능하다. 첫 번째는 소프트웨어적인 방법으로 기존의 데이터 메모리 공간에 스택의 영역을 할당해 사용하는 방법이 있으며 두 번째로 스택을 하드웨어로 직접 구현하여 칩(chip)속에 내장시켜 사용하는 방법이 있다.

소프트웨어적인 방법은 스택의 크기를 사용자가 원하는 대로 사용할 수 있기 때문에 사용자가 쓰기에 편리하다. 그러나, 메모리를 스택으로 사용하기 때문에 하드웨어적인 방법보다 실행 속도가 느리고, 전력 소모가 많다는 단점이 있다.

하드웨어적인 방법은 일반적으로 스택의 크기가 작고 사용할 수 있는 크기가 고정되므로 사용자가 사용할 수 있는 범위가 제한되나 콘텍스트 스위칭(context switching)과 같이 다량의 스택을 동시에 사용하는 경우 속도가 빠를뿐만 아니라 외부 메모리를 액세스하지 않기 때문에 전력 소모 양이 적다는 장점이 있다. 따라서 저 전력이 요구되는 마이크로 프로세서 유닛이나 디지털 신호 처리기에서는 일

반적으로 하드웨어적인 스택 구조를 갖는다.

8 비트(bit) 마이크로 컨트롤러 유닛의 경우 처리하는 데이터의 단위는 8 비트인데 반해 어드레스(address)는 16 비트 또 그 이상이 된다. 이는 어드레스로 8 비트만을 사용할 경우 최대 프로그램의 크기가 256 이므로 원하는 프로그램을 저장하기에는 부족하기 때문이다. 따라서, 대부분의 8 비트 마이크로 컨트롤러 유닛은 어드레스의 비트 폭(bit width)을 16 비트 또는 20 비트를 사용하게 된다.

마이크로 컨트롤러 유닛 내에서 처리되는 데이터와 어드레스의 비트 폭이 다른 경우 스택을 사용하는 데에 있어서 문제가 발생한다. 즉, 내부의 레지스터(register) 값을 스택에 저장하거나(PUSH), 스택에서 가져올 경우(POP)에는 스택이 8 비트로 동작해야 하는데 반해 어드레스 값을 스택에 저장하거나(CALL, interrupt), 스택에서 가져올 경우(Return)에는 어드레스의 비트 폭인 16 비트 또는 20 비트로 동작하여야 한다.

이와 같은 문제점을 해결하기 위해 일반적으로 사용되는 방법은 다중 뱅크 구조(multi-bank structure)의 스택을 사용하는 것이다. 다중 뱅크 구조란 8비트 데이터 스토레지(storage)가 순차적으로 액세스(access) 되는 단일 뱅크 구조와는 달리 8비트 데이터 스토레지를 여러 개의 뱅크로 나누어 각각에 대하여 동시에 액세스 될 수 있는 구조를 말한다. 이와 같은 다중 뱅크 구조의 스택을 사용하면 단일 뱅크 구조에서 여러 번의 동작이 필요한 어드레스 값의 저장을 단지 한번의 동작으로 수행할 수 있다. 즉 단일 뱅크 구조에서는 어드레스의 크기가 16 비트일 경우 이를 상위와 하위 8 비트로 나누어 두 번에 나눠서 저장(PUSH) 및 삭제(POP)를 해야

하는데 반해 다중 뱅크 구조에서는 2 개의 뱅크에 동시에 값을 저장하거나 송출할 수 있다. 그 결과, 어드레스 값의 저장 및 삭제 과정이 단축되어 보다 빠르게 처리할 수 있다는 장점을 갖는다.

도 1은 종래의 다중 뱅크 스택을 제어하기 위한 회로를 블록적으로 보여주고 있다.

도 1을 참조하면, 다중 뱅크 스택을 제어하기 위한 회로는 스택 포인터(stack pointer)(10)와, 가산 및 감산을 수행하기 위한 가감산기(20), 상기 스택 포인터(10) 및 가감산기(20)의 출력을 입력하여 선택적으로 출력하기 위한 멀티플렉서(30)와, 증가기(40), 상기 멀티플렉서(30)와 증가기(40)의 출력을 선택적으로 출력하기 위한 멀티플렉서들(50, 60)과, 뱅크 1의 스택 포인터(70) 및 뱅크 2의 스택 포인터(80)로 구성된다.

2 개의 뱅크(뱅크0, 뱅크 1)에 대한 각각의 포인터를 만들기 위해서는 우선 스택을 액세스하는 두 개의 연속적인 포인터(상위 포인터, 하위 포인터)를 만들어야 한다. 이는 8비트의 데이터뿐만 아니라 16비트의 어드레스 값을 처리하기 위해서이다. 상기 포인터들을 만들기 위해서는 +1, +2, -1, -2 등의 가산 및 감산을 수행해야 하는데 이는 가감산기(20)에서 수행된다. +1은 데이터를 스택에 저장할 때(PUSH 명령어) 수행되고, -1은 데이터를 스택에서 삭제할 때(POP) 수행된다. 또, +2는 어드레스를 스택에 저장할 때(CALL) 수행되고, -2는 어드레스를 스택에서 삭제(RETURN)할 때에 수행된다. 이 값에서 상위 포인터를 만들기 위해서는 증가기(40)에서 +1 만큼 증가시킨다. 멀티플렉서들(50, 60)은 상기 상위 포인터와 하위 포인터 가운에 하나를 각각의 뱅크에 선택하여 스택 포인터들(70, 80)에 제공한다. 이

것은 현재 스택의 최상위 데이터가 뱅크 0 또는 뱅크 1에 있느냐에 따라 각각의 뱅크에 인가되는 포인터가 달라지기 때문이다.

상술한 바와 같이, 다중 뱅크 스택을 제어하기 위해서는 각각의 뱅크(Bank 0, Bank 1)에 대해 스택 포인터(70, 80)를 제어해야 하므로 제어 로직이 복잡해진다. 스택이 다중 뱅크 구조를 갖더라도 사용자는 하나의 스택 포인터(10)를 사용할 수 있어야 하므로 다중 뱅크 구조에서는 사용자 스택 포인터 값을 이용하여 각 뱅크의 스택 포인터(70, 80) 값을 계산해야 한다. 또한 단일 뱅크 구조에서는 스택이 한 사이클에 항상 1 만큼씩만 증가 또는 감소하므로 덧셈기(Adder)가 아닌 증감기(incrementer)만으로 스택 포인터 값을 변화시킬 수 있지만 다중 뱅크 구조에서는 2 만큼 증가 또는 감소할 수 있으므로 덧셈기를 사용해야 한다. 따라서 하드웨어 구성이 복잡해진다.

【발명이 이루고자 하는 기술적 과제】

따라서, 본 발명의 목적은 상술한 제반 문제점을 해결하기 위해 제안된 것으로, 다중 뱅크 스택을 효율적으로 제어할 수 있는 제어 회로를 제공하는데 있다.

【발명의 구성 및 작용】

상술한 바와 같은 본 발명의 목적을 달성하기 위한 본 발명의 특징에 의하면, 폭(width)이 M 비트이고 최대 2^N 개를 저장할 수 있는 스택(stack)을 K 개의 다중 뱅크(multi-bank) 구조로 사용하기 위한 제어 회로는: 상기 스택의 현재 주소 값을 저장하기 위한 제 1 스택 포인터와; 상기 K 개의 다중 뱅크 각각의 현재 주소 값을 저장하기 위한 K 개의 제 2 스택 포인터들과; 상기 제 1 스택 포인터에 저장된 주

소 값의 상위 X 비트에 1을 가/감산하여 출력하는 수단과; 상기 제 1 스택 포인터에 저장된 주소 값과 상기 가/감산 수단의 연산 결과를 입력하여 둘 중의 하나를 선택적으로 상기 제 1 스택 포인터로 제공하는 제 1 선택 수단 및; 상기 K 개의 제 2 스택 포인터들에 각각 대응하고, 상기 제 1 스택 포인터에 저장된 주소 값과 상기 가/감산 수단의 연산 결과를 입력하여 둘 중의 하나를 선택적으로 상기 제 2 스택 포인터들로 제공하는 K 개의 제 2 선택 수단들을 포함하되, 상기 가/감산 수단, 선택 수단, 제 1 및 제 2 선택 수단들은 현재 수행되는 명령어의 종류에 따라 제어되고,

상기 제 1 스택 포인터의 하위 N - X 비트는 현재 주소 값에 대응하는 뱅크를 표시하고, 현재 수행되는 명령어에 의해 M 비트의 데이터/어드레스가 상기 스택에 저장(push)되거나 인출(pop)될 때 토글 된다.

바람직한 실시예에 있어서, 상기 제 1 및 제 2 선택 수단들은 멀티플렉서(multiplexor)로 구성된다.

(실시예)

이하 본 발명에 따른 실시예를 첨부된 도면 도 2 내지 도 5를 참조하여 상세히 설명한다.

도 2는 일반적인 스택의 구조를 블록적으로 보여주고 있다.

도 2에 도시된 바와 같이, 스택(100)은 비트 폭은 8 비트이고 최대 32 개의 값을 저장할 수 있는 32 워드(word)의 크기를 갖는다. 스택 포인터(stack pointer; SP)(110)는 스택의 어드레스(address)가 0 번지부터 31 번지까지 지정할 수 있어야

하므로 5비트의 크기를 갖는다. 상기 스택 포인터(110)는 항상 데이터 값이 저장된 최상위 어드레스에 1을 더한 곳(Top of Stack; TOS)을 가리키고 있다.

도 3은 도 2에 도시된 스택을 다중 뱅크 구조로 분할하여 사용할 때 스택의 구조를 블록적으로 보여주는 도면이다.

도 3에 도시된 스택(200)은 어드레스의 최하위 비트가 0인 것을 뱅크 0(Bank 0)으로, 어드레스의 최하위 비트가 1인 것을 뱅크 1(Bank 1)로 나눈 것이다. 스택 포인터(110)는 항상 데이터 값이 저장된 최상위 어드레스에 1을 더한 곳을 가리킨다. 뱅크 0의 스택 포인터(Bank 0 SP)(230)는 뱅크 0에서 데이터 값이 저장된 최상위 어드레스에 1을 더한 곳을 가리키고, 뱅크 1의 스택 포인터(Bank 1 SP)(220)는 뱅크 1에서 데이터 값이 저장된 최상위 어드레스에 1을 더한 곳을 가리킨다.

도 4a 및 도 4b는 도 3에 도시된 다중 뱅크 구조의 스택을 각각 뱅크 0과 뱅크 1로 나누어 도시한 도면이다.

우선 도 4a를 참조하면, 뱅크 1의 스택(200a)은 도 3에 도시된 스택(200)의 최하위 비트가 1인 것만을 모아 나타낸 것으로, 최하위 비트를 제외하여 어드레스를 0 번지부터 15번지까지로 표시하였다. 따라서 스택 포인터(220)는 4 비트가 된다.

도 4b는 뱅크 0의 스택(200b)을 도시하고 있다. 상기 스택(200b)은 도 3에 도시된 스택(200)의 최하위 비트가 0인 것만을 모아 나타낸 것으로, 최하위 비트를 제외하여 어드레스를 0 번지부터 15번지까지로 표시하였다. 뱅크 0의 스택 포인터(230) 또한 4 비트가 된다.

도 5는 본 발명의 바람직한 실시예에 따른 다중 뱅크 구조의 스택을 제어하기 위한

회로를 블록적으로 도시한 도면이다.

도면에 도시된 바와 같이, 다중 뱅크 구조의 스택을 제어하기 위한 회로는 스택 포인터(210)의 값을 +1 또는 -1 만큼 가감하기 위한 가감산기(incrementer/decrementer)와, 멀티플렉서(multiplexor)들(214, 216, 218)로 구성된다. 상기 멀티플렉서 1(214)의 출력은 스택 포인터(210)로 제공되고, 멀티플렉서 2 및 3(216, 218)의 출력은 각각 뱅크 1 및 뱅크 0의 스택 포인터들(220, 230)로 입력된다.

다시 도 5를 참조하면, 5 비트의 스택 포인터(210)는 상위 4 비트와 하위 1 비트로 나뉘어 상위 4비트는 가감산기(212)와 멀티플렉서(214)로 입력되고, 하위 1 비트는 현재 수행되는 명령어의 종류에 따라 그 값이 토글(toggle)되거나 변하지 않는다. 멀티플렉서 2 및 멀티플렉서 3(216, 218)은 상기 스택 포인터(210)의 상위 4 비트와 가감산기(212)의 출력을 입력하여 현재 수행되는 명령어에 의해 두 입력 가운데 하나를 각각 뱅크 1 및 뱅크 0의 스택 포인터들(220, 230)로 제공한다.

다음 [표 1] 및 [표 2]를 참조하여 제어회로의 동작과 스택 포인터의 변화를 상세히 설명한다.

[표 1]

Instruction	Bank 1 SP	Bank 0 SP	SP [4:1]	SP [0]
PUSH	SP	SP	SP	toggle
CALL	SP	SP	SP + 1	-
POP	SP - 1	SP - 1	SP - 1	toggle
RETURN	SP - 1	SP - 1	SP - 1	-

우선 [표 1]을 참조하면, 상기 [표 1]은 스택 포인터(210)의 최하위 비트가 0 일 때, 스택에 데이터 또는 어드레스를 저장하거나 삭제하는 명령어들에 의해 스택 포인터들(210, 220, 230)의 변화를 나타낸 것이다. 상기 스택 포인터(210)의 최하위 비트의 값이 0일 경우에는 현재 스택의 뱅크 1까지 데이터가 저장되어 있다는 것을 의미한다.

각 명령어(instruction)에 대한 스택 포인터의 변화는 다음과 같다. 먼저, 스택에 데이터를 저장할 경우(PUSH), 뱅크 1 및 뱅크 0의 스택 포인터들(220, 230)에는 스택 포인터(210)의 상위 4 비트 값이 입력되고 스택 포인터(210)의 최하위 비트는 토글 된다. 스택에 16비트의 어드레스를 저장할 경우(CALL)에는 뱅크 1 및 뱅크 0의 스택 포인터들(220, 230)에는 스택 포인터(210)의 상위 4 비트 값이 입력되고, 스택 포인터(210)의 상위 4비트는 가감산기(212)에서 1 더해진 값이 저장되고, 최하위 비트는 그대로 유지된다.

스택에 저장된 데이터를 삭제하는 경우(POP), 뱅크 1 및 뱅크 0의 스택 포인터들(220, 230)에는 스택 포인터(210)의 상위 4 비트 값에서 1을 감산한 값이 입력되고 스택 포인터(210)의 최하위 비트는 토글 된다. 스택에 저장된 어드레스를 삭제하

는 경우(RETURN), 뱅크 1 및 뱅크 0의 스택 포인터들(220, 230)과 스택 포인터(210)의 상위 4비트에는 가감산기(212)에서 1이 감산된 값이 입력된다. 스택 포인터(210)의 최하위 비트는 그대로 유지된다.

[표 2]

Instruction	Bank 1 SP	Bank 0 SP	SP [4:1]	SP [0]
PUSH	SP	SP + 1	SP + 1	toggle
CALL	SP	SP + 1	SP + 1	-
POP	SP - 1	SP	SP	toggle
RETURN	SP - 1	SP	SP - 1	-

[표 2]는 스택 포인터(210)의 최하위 비트가 1 일 때, 스택에 데이터 또는 어드레스를 저장하거나 삭제하는 명령어들에 의해 스택 포인터들(210, 220, 230)의 변화를 나타낸 것이다. 상기 스택 포인터(210)의 최하위 비트의 값이 1일 경우에는 현재 스택의 뱅크 0까지 데이터가 저장되어 있다는 것을 의미한다. 이러한 경우, 뱅크 0의 스택 포인터(220) 값은 뱅크 1의 스택 포인터(230)보다 1 만큼 큰 값을 갖는다. 왜냐하면, 스택에 가장 위에 저장된 값이 뱅크 0에 있으므로 뱅크 0은 그 다음 주소를 가리켜야 하는데 반해 뱅크 1은 아직 그 어드레스에 값이 씌여지지 않았기 때문이다.

각 명령어(instruction)에 대한 스택 포인터의 변화는 다음과 같다. 먼저, 스택에 데이터나 어드레스를 저장할 경우(PUSH, CALL), 뱅크 1의 스택 포인터(220)에는 스택 포인터(210)의 상위 4 비트 값이 입력되고, 뱅크 0의 스택 포인터(230)와 스택 포인터(210)에는 가감산기(212)에서 1 더해진 값이 입력된다. 스택에 저장된 데이

터를 삭제하는 경우(POP) 뱅크 1의 스택 포인터(220)에는 가감산기(212)에서 1이 감산된 값이 입력되고, 뱅크 0의 스택 포인터(230)와 상기 스택 포인터(210)에는 상기 스택 포인터(210)의 값이 그대로 입력된다. 스택에 저장된 어드레스를 삭제하는 경우(RETURN), 뱅크 0의 스택 포인터(230)에는 스택 포인터(210)의 값이 입력되고, 뱅크 1의 스택 포인터(220)와 스택 포인터(210)의 상위 4 비트에는 가감산기(212)에서 1이 감산된 값이 입력된다.

스택 포인터(210)의 최하위 비트가 1인 경우, 스택에 8비트의 데이터를 저장하거나 삭제할 때 스택 포인터(210)의 최하위 비트는 토글되고, 스택에 16비트의 어드레스를 저장하거나 삭제할 때 스택 포인터(210)의 최하위 비트는 그대로 유지된다.

상술한 바와 같이, 본 발명의 다중 뱅크 구조의 스택을 제어하기 위한 회로는 스택 포인터(210)의 상위 4비트에 1을 가산 또는 감산하기 위한 가감산기(212)와, 상기 스택 포인터(210) 및 뱅크 스택 포인터들(220, 230)로 포인터를 선택적으로 제공하는 세 개의 멀티플렉서(214, 216, 218)로 구성된다. 각 뱅크의 스택 포인터와 업데이트(update)될 스택 포인터의 값이 동시에 +1 또는 -1로 변경되는 경우가 없다는 특성을 이용하여 하나의 가감산기만으로 각 뱅크에 대한 포인터 값을 구할 수 있다. 또, 전체 스택의 크기는 32 이지만 각 뱅크의 크기는 16 이므로, 스택 포인터의 전체 5 비트를 이용하지 않고 상위 4 비트를 각 뱅크의 스택 포인터로 하고, 최하위 비트는 뱅크를 선택하는데 이용한다. 스택의 어드레스를 지정하기 위해 5 비트 전체를 사용할 경우에는 +1, +2, -1, -2를 하기 위해 5 비트의 가감산기가 필요하나 상위 4 비트만을 이용할 경우, +1, -1 만을 수행할 수 있으면 되므로 4 비

트 가감산기로 구현이 가능하다.

이상에서, 본 발명에 따른 회로의 구성 및 동작을 상기한 설명 및 도면에 따라 도시하였지만 이는 예를 들어 설명한 것에 불과하며 본 발명의 기술적 사상을 벗어나지 않는 범위 내에서 다양한 변화 및 변경이 가능함은 물론이다.

【발명의 효과】

이상과 같은 본 발명에 의하면, 다중 뱅크 구조의 스택을 제어하기 위한 회로는 종래의 제어 회로보다 회로 구성이 간단하므로 하드웨어 구성 면적이 줄어든다. 또한, 소비 전력이 감소된다.

【특허청구범위】

【청구항 1】

폭(width)이 M 비트이고 최대 2^N 개를 저장할 수 있는 스택(stack)을 K 개의 다중 뱅크(multi-bank) 구조로 사용하기 위한 제어 회로에 있어서,

상기 스택의 현재 주소 값을 저장하기 위한 제 1 스택 포인터와;

상기 K 개의 다중 뱅크 각각의 현재 주소 값을 저장하기 위한 K 개의 제 2 스택 포인터들과;

상기 제 1 스택 포인터에 저장된 주소 값의 상위 X 비트에 1을 가/감산하여 출력하는 수단과;

상기 제 1 스택 포인터에 저장된 주소 값과 상기 가/감산 수단의 연산 결과를 입력하여 둘 중의 하나를 선택적으로 상기 제 1 스택 포인터로 제공하는 제 1 선택 수단 및;

상기 K 개의 제 2 스택 포인터들에 각각 대응하고, 상기 제 1 스택 포인터에 저장된 주소 값과 상기 가/감산 수단의 연산 결과를 입력하여 둘 중의 하나를 선택적으로 상기 제 2 스택 포인터들로 제공하는 K 개의 제 2 선택 수단들을 포함하되,

상기 가/감산 수단, 선택 수단, 제 1 및 제 2 선택 수단들은 현재 수행되는 명령어의 종류에 따라 제어되고,

상기 제 1 스택 포인터의 하위 N - X 비트는 현재 주소 값에 대응하는 뱅크를 표시하고, 현재 수행되는 명령어에 의해 M 비트의 데이터/어드레스가 상기 스택에 저장(push)되거나 인출(pop)될 때 토글되는 것을 특징으로 하는 다중 뱅크(multi-bank)

스택 구조를 위한 제어 회로.

【청구항 2】

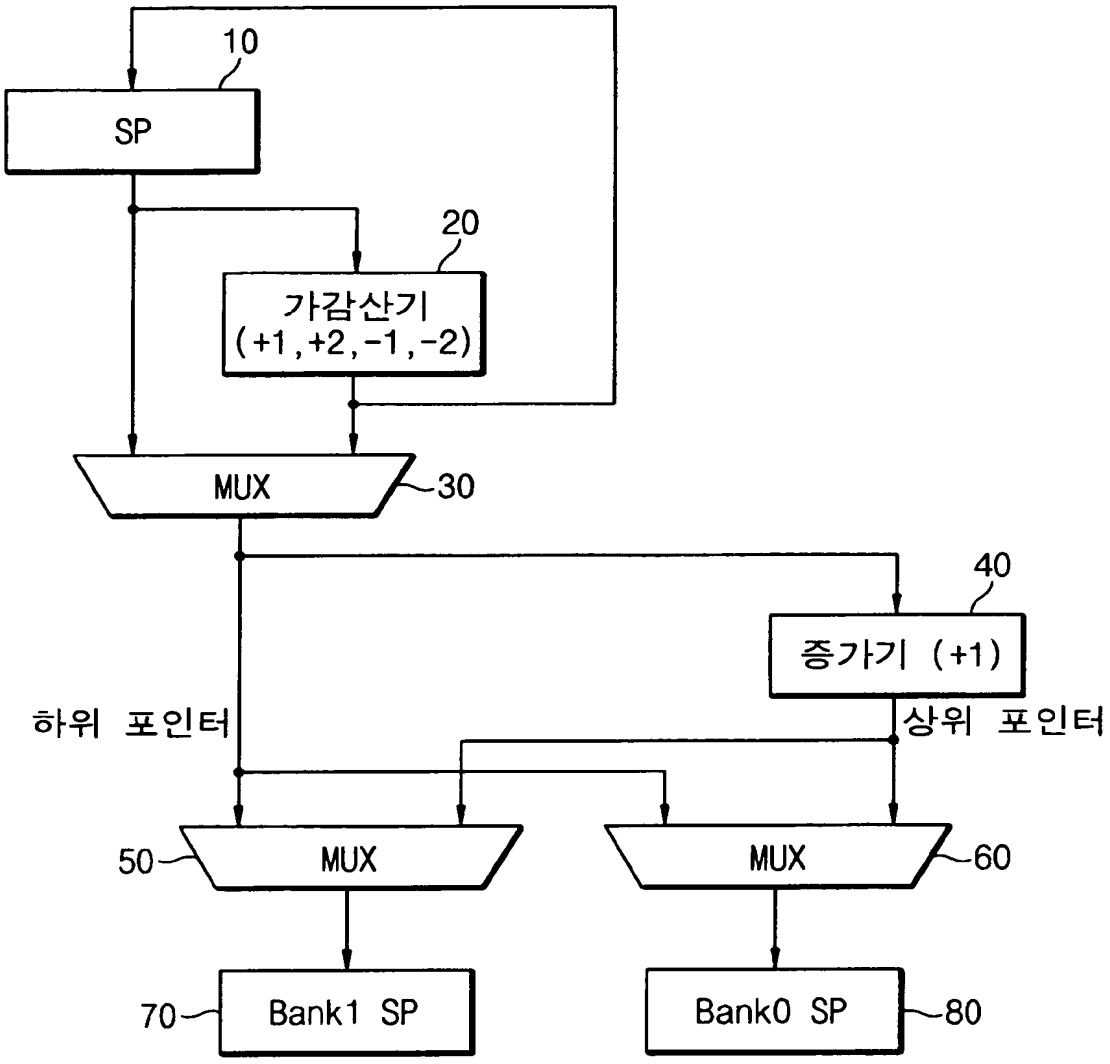
제 1 항에 있어서,

상기 제 1 및 제 2 선택 수단들은 멀티플렉서(multiplexor)인 것을 특징으로 하는 다중 뱅크(multi-bank) 스택 구조를 위한 제어 회로.

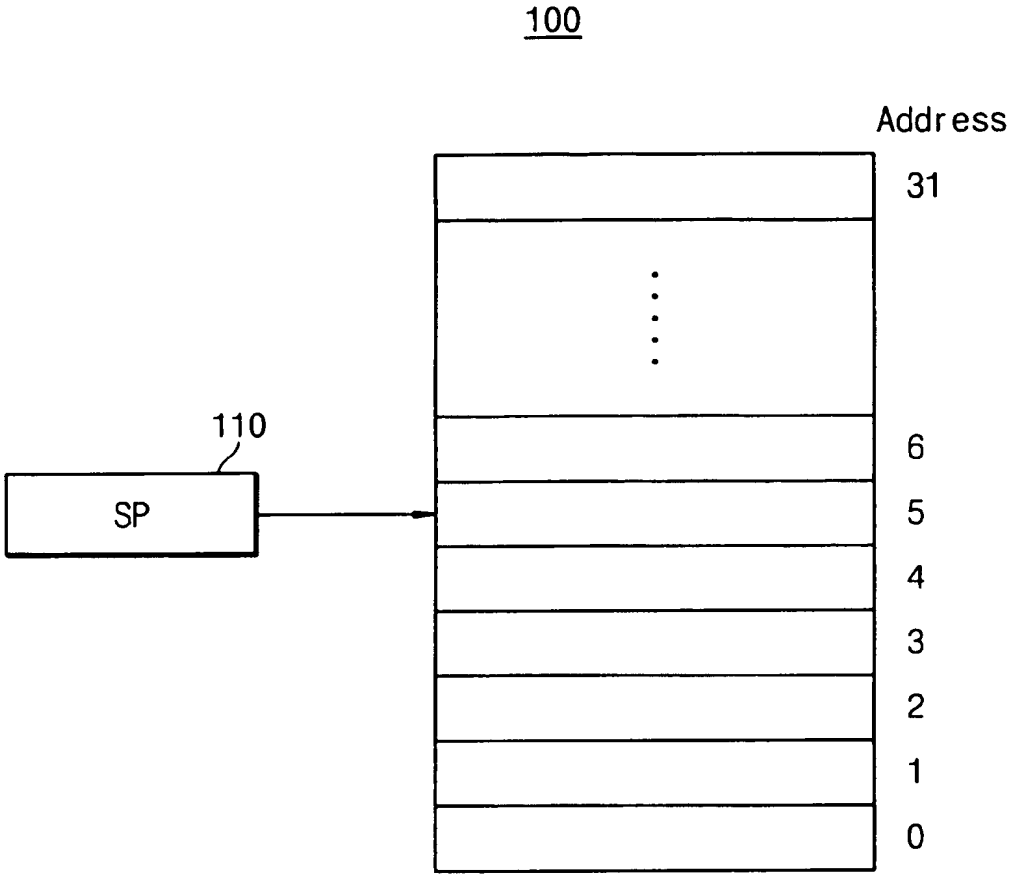
【도면】

【도 1】

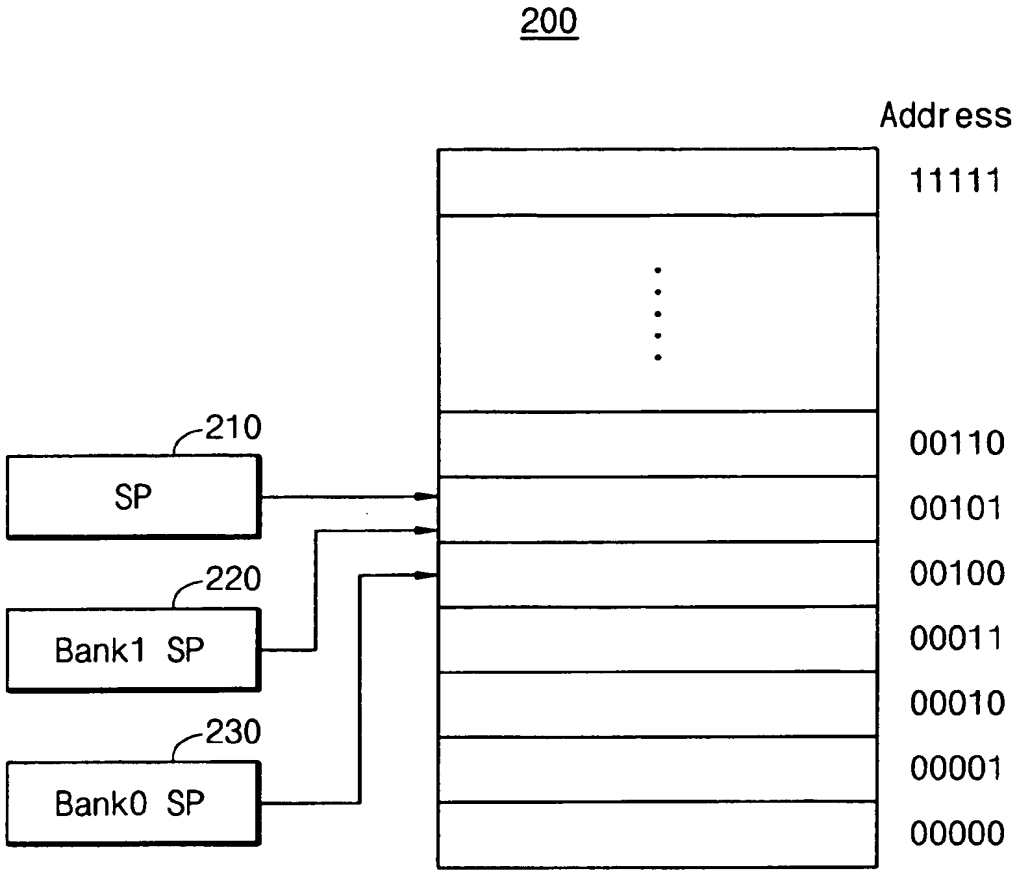
(종래기술)


10
SP
20
가감산기
(+1,+2,-1,-2)
MUX
30
40
증가기 (+1)
하위 포인터
상위 포인터
50
MUX
MUX
60
70
Bank1 SP
Bank0 SP
80

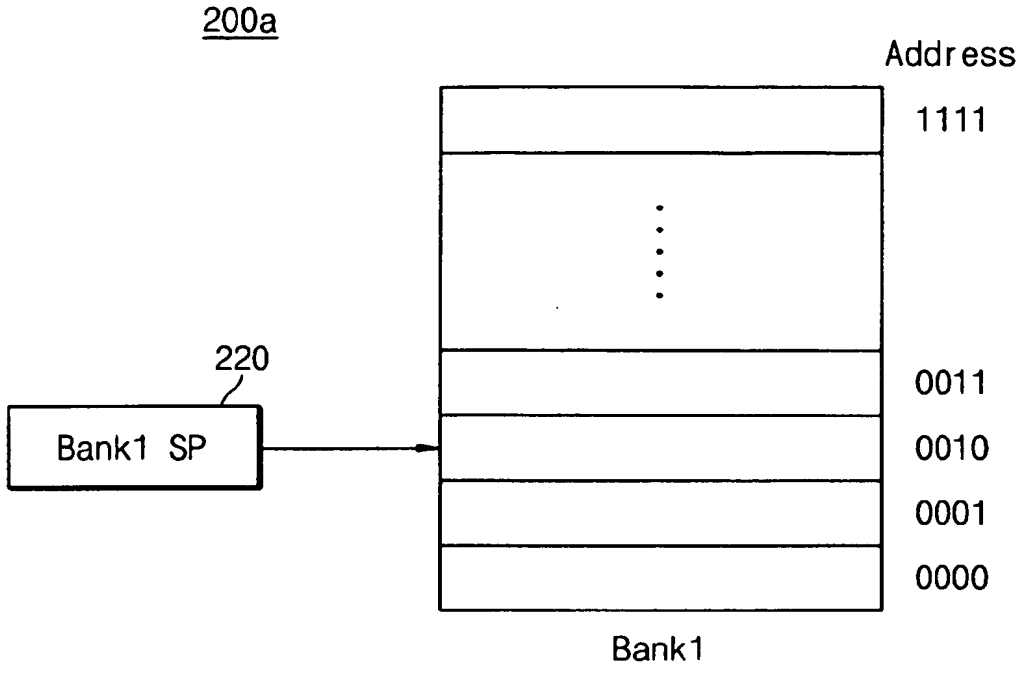
【도 2】



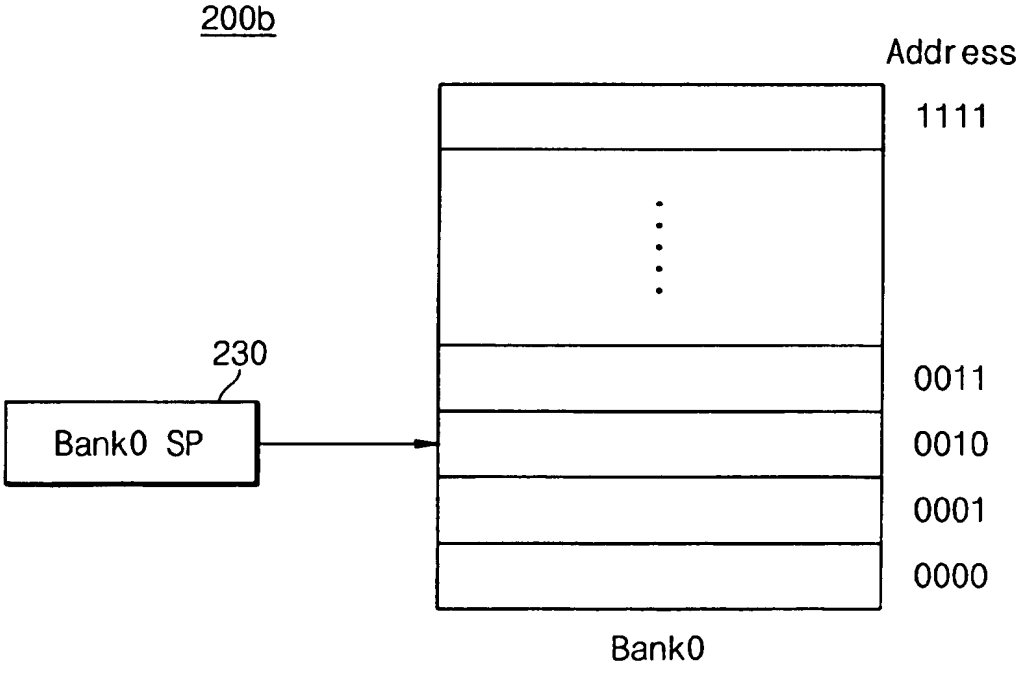
【도 3】


200
Address
11111
00110
00101
00100
00011
00010
00001
00000
210
SP
220
Bank1 SP
230
Bank0 SP

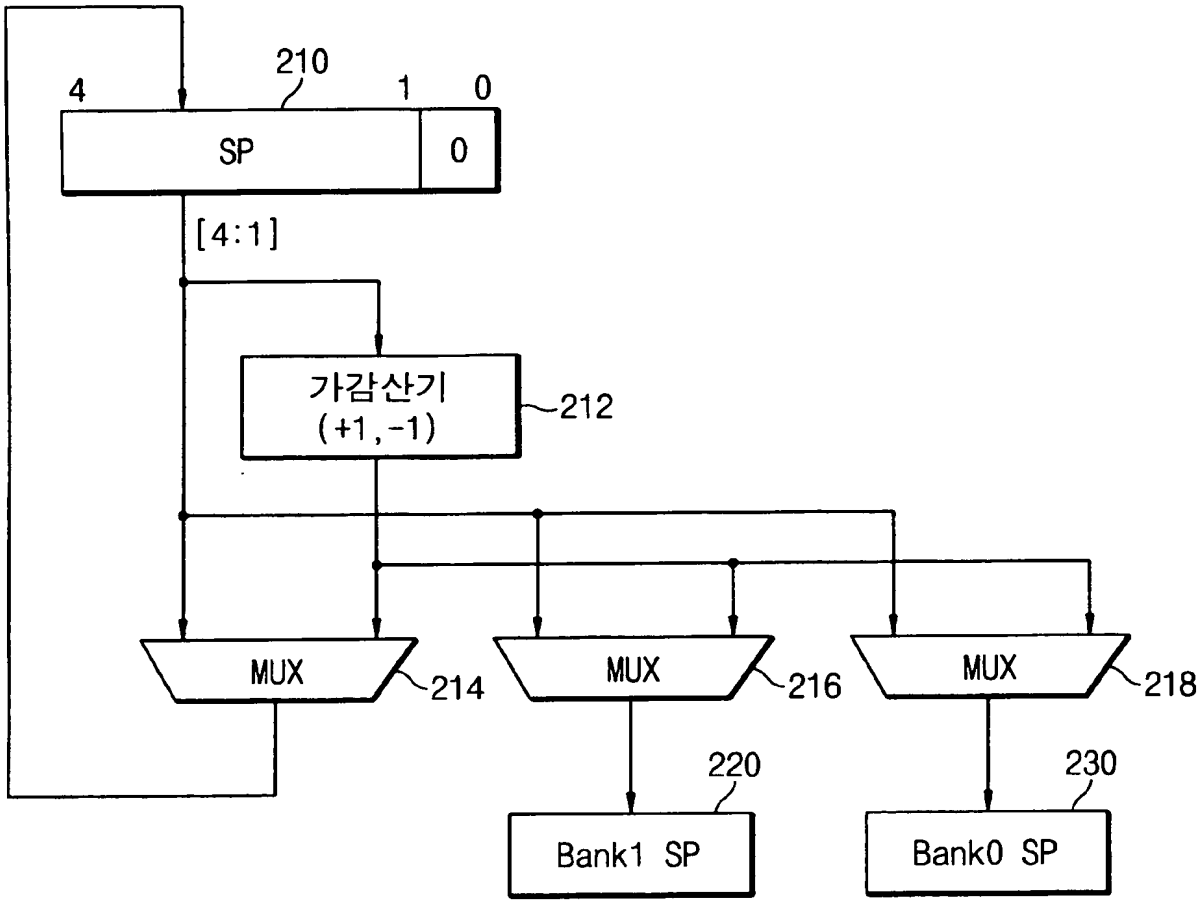
【도 4a】


200a
Address
1111
220
Bank1 SP
0011
0010
0001
0000
Bank1

【도 4b】


200b
Address
1111
230
Bank0 SP
0011
0010
0001
0000
Bank0

【도 5】


210
4
1
0
SP
0
[4:1]
가감산기
(+1,-1)
212
MUX
214
MUX
216
MUX
218
220
230
Bank1 SP
Bank0 SP